Chapter 1

Product Distributions for Distributed Optimization

Stefan R. Bieniawski
Department of Aeronautics & Astronautics
Stanford University
stefanb@stanford.edu
David H. Wolpert
NASA Ames Research Center
dhw@ptolemy.arc.nasa.gov

With connections to bounded rational game theory, information theory and statistical mechanics, Product Distribution (PD) theory provides a new framework for performing distributed optimization. Furthermore, as an extension of Collective Intelligence, PD theory connects distributed optimization to distributed Reinforcement Learning (RL). This paper provides an overview of PD theory and details an algorithm for performing optimization derived from it. The approach is demonstrated on two unconstrained example optimization problems, one with discrete variables and one with continuous variables. To highlight the connections between PD theory and distributed RL, the results are compared with those obtained with distributed reinforcement learning inspired optimization approaches. The inter-relationship of the techniques is discussed.

1.1 Introduction

Traditional optimization techniques use centralized approaches for obtaining a solution to a problem. Although significant effort has been made to make the techniques applicable to large, sparsely connected problems, an alternate approach pursued here is to distribute the optimization among agents that rep-

resent the variables in the problem. Formulating the problem as a distributed stochastic optimization allows for the application of techniques from machine learning, statistics, multi-agent systems, and game theory. In addition, approaches can be developed using these fields which take advantage of the sparse, local structure of certain optimization problems. The current work leverages the aforementioned fields by applying a Collective Intelligence (COIN) technique, Product Distribution (PD) theory, to several example optimization problems. In this approach, the optimization becomes finding the equilibrium of a system whose agents are the variables and whose world utility is the objective function.

Typically in stochastic optimization approaches probability distributions are used to help search for a point in the variable space which optimizes the objective function. In contrast, in the PD approach the search is for a probability distribution across the variable space that optimizes an associated Lagrangian. Since the probability distribution is a vector in a Euclidean space, the search can be done via gradient based methods even if the variable space is categorical. Similar techniques have been successfully applied to a variety of distributed optimization problems including network routing, computing resource allocation, and data collection by autonomous rovers [2, 1].

PD theory can be viewed as the information-theoretic extension of conventional full-rationality game theory to the case of bounded rational agents [3]. Information theory shows that the equilibrium of a game played by bounded rational agents is the optimizer of a Lagrangian of the probability distribution of the agents' joint-moves. In any game, bounded rational or otherwise, the agents are independent, with each agent i choosing its move x_i at any instant by sampling its probability distribution (mixed strategy) at that instant, $q_i(x_i)$. Accordingly, the distribution of the joint-moves is a product distribution, $P(x) = \prod_i q_i(x_i)$. In this representation, all coupling between the agents occurs indirectly; it is the separate distributions of the agents $\{q_i\}$ that are coupled, while the actual moves of the agents are independent. As a result the optimization of the Lagrangian can be done in a completely distributed manner. This approach provides a broadly applicable way to cast any constrained optimization problem as the equilibrating process of a multi-agent system, together with an efficient method for that equilibrating process.

The next section presents the approaches for minimizing the Lagrangian provided by PD theory and the related approaches obtained from distributed RL. The performance on several example problems is then presented.

1.2 Optimizing the Lagrangian

Given that the agents in a multi-agent system are bounded rational, if they play a team game with world utility G, their equilibrium will be the optimizer of G. The equilibrium can be found by minimizing a Lagrangian which is a function of the agents' probabilities [3]. Specifically, for the unconstrained optimization

problem,

$$\min_{\vec{x}} G(\vec{x})$$

assume each agent sets one component of $\vec{x}$ as that agent's action. The Lagrangian $\mathcal{L}_i(q_i)$ for each agent as a function of the probability distribution across its actions is,

$$\begin{aligned}\mathcal{L}_i(q_i) &= E[G(x_i, x_{(i)})] \; - \; T\, S(q_i) \\ &= \sum_{x_i} q_i(x_i) E[G(x_i, x_{(i)})|x_i] \; - \; T\, S(q_i)\end{aligned}$$

where G is the world utility (system objective) which depends upon the action of agent i, x_i, and the actions of the other agents, $x_{(i)}$. The expectation $E[G(x_i, x_{(i)})|x_i]$ is evaluated according to the distributions of the agents other than i:

$$P(x_{(i)}) = \prod_{j \neq i} q_j(x_j)$$

The entropy S is given by:

$$S(q_i) = -\sum_{x_j} q_i(x_j)\, ln\, q_i(x_j)$$

Each agent then addresses the following local optimization problem,

$$\min_{q_i} \mathcal{L}_i(q_i)$$

$$\text{s.t.} \quad \sum_{x_i} q_i(x_i) = 1, \quad q_i(x_i) \geq 0, \forall x_i$$

The Lagrangian is composed of two terms weighted by the temperature T: the expected reward across i's actions, and the entropy associated with the probability distribution across i's actions. During the minimization of the Lagrangian, the temperature provides the means to trade-off exploitation of good actions (low temperature) with exploration of other possible actions (high temperature).

In this paper two algorithms for optimizing the Lagrangian are considered. The first is a variant of Newton's method for directly descending the Lagrangian. The second is Brouwer updating, which, under different names, is perhaps the most common scheme employed in RL-based algorithms for finding game equilibria.

1.2.1 Nearest Newton descent

The minimization of the Lagrangian is amenable to solution using gradient descent or Newton updating since both the gradient and the Hessian are obtained in closed form. Using Newton updating and enforcing the constraint on total

probability, the following update rule, referred to as Nearest Newton [3, **?**], is obtained:

$$q_i(x_i) \rightarrow q_i(x_i) - \alpha q_i(x_i) \times \left\{ \frac{E[G|x_i] - E[G]}{T} + S(q_i) + \ln q_i(x_i) \right\} \tag{1.1}$$

where α plays the role of a step size. The step size is required since the expectations result from the current probability distributions.

1.2.2 Role of private utilities

Performing the update at each iteration involves a separate conditional expected utility for each agent. These are estimated either exactly if a closed form expression is available or with Monte-Carlo sampling if no simple closed form exists. Since accurate estimates usually require extensive sampling, the G occurring in each agent i's update rule can be replaced with a private utility g_i chosen to ensure that the Monte Carlo estimation of $E(g_i|x_i)$ has both low bias (with respect to estimating $E(G|x_i)$ and low variance [7]. Intuitively bias represents the alignment between the private utility and world utility. With zero bias, updates which reduce the private utility are guaranteed to also reduce the world utility. It is also desirable for an agent to distinguish its contribution from that of the other agents: variance measures this sensitivity. With low variance, the agents can perform the individual optimizations accurately without a large number of Monte-Carlo samples.

Two private utilities are used for the example problems in this work, Team Game (TG) and Wonderful Life Utility (WLU)[3, **?**, 2, 1]. These are defined as:

$$g_{TG_i}(x_i, x_{(i)}) = G(x_i, x_{(i)})$$

$$g_{WLU_i}(x_i, x_{(i)}) = G(x_i, x_{(i)}) - G(CL_i, x_{(i)})$$

For the team game, the private utility is simply the world utility. For WLU, the private utility is the world utility minus the world utility with the agent action "clamped" by the value CL_i. Both of these utilities have zero bias. However, due to the subtracted term, WLU has much lower variance than TG.

1.2.3 Brouwer updating

An alternate way to try to find the q that minimizes the Lagrangian is an iterative process akin to the best-response scheme of game theory [8]. Given any current distribution q, all agents i simultaneously replace their current distributions. In this replacement each agent i replaces q_i with the distribution given by More precisely the update rule becomes,

$$q_i(x_i) = e^{-\frac{1}{T} E_{q_{(i)}}[g_i|x_i]} / \sum_{x_i} e^{-\frac{1}{T} E_{q_{(i)}}[g_i|x_i]} \tag{1.2}$$

where the expectations are based on the current $q_{(i)}$. The conditional expected utilities can, if necessary, be estimated once again using Monte-Carlo sampling. One problem with Brouwer updating is that there is no reason to believe that it will converge. In practice the Monte Carlo samples are "aged" by a factor γ, to weight older sample points less heavily than more recent points. See [2, 1] for details. This modification to Brouwer updating still provides no formal guarantees although it improves performance. Such guarantees are obtained, however, if rather than conventional "parallel" Brouwer updating, one uses "serial Brouwer updating", in which only one agent at time updates its distribution.

1.3 Results

1.3.1 Discrete optimization example problem

A discrete optimization problem is used to compare the methods for minimizing the Lagrangian. Specifically, the bin packing problem from the operations research literature was selected [4]. This problem consists of assigning N items (the agents) of differing sizes into the smallest number of bins each with capacity c. For the current study instances were chosen which have a designed minimum number of bins and were obtained from the OR-Library [4]. The instances consisted of 60 items to be packed in groups of three into 20 bins each of capacity 100. Since in general the minimum number of bins is not known, the move space of the agents was set to the number of items. The objective function used is,

$$G = \begin{cases} \sum_{i=1}^{N} [(\frac{c}{2})^2 - (x_i - \frac{c}{2})^2] & if\ x_i \leq c \\ \sum_{i=1}^{N} (x_i - \frac{c}{2})^2 & if\ x_i > c \end{cases} \tag{1.3}$$

where x_i is the total size of the items in bin i. This form of the objective function encourages either full or empty bins and strongly penalizes overfilled bins. Although PD theory can be extended to explicitly include constraints, the penalty function formulation in (1.3) allows for a more direct comparison of the approaches.

A problem single variant was selected and 20 cases used to evaluate the performance of the approaches. Figure 1.1 compares all three schemes. Shown is the average objective as a function of iteration. The greedy serial Brouwer approach updates only the agent causing the largest decrease in the Lagrangian at each iteration. For these results WLU is used with the clamping value set to 0, removing that item from the objective evaluation. The remaining parameter settings are $\alpha = 0.1$, $\gamma = 0.8$, and 100 Monte-Carlo samples for each block. The Nearest Newton approach and parallel Brouwer are seen to obtain similar results, with Nearest Newton converging faster. Both tend to find solutions which are one to two bins over the minimum required. Greedy serial Brouwer tends to obtain poorer solutions due to the combination of limited sampling and overly greedy updating. Since only a single agent is updated each iteration the convergence is also quite slow, particularly for the greedy implementation where the agents must all evaluate evaluate their private utilities.

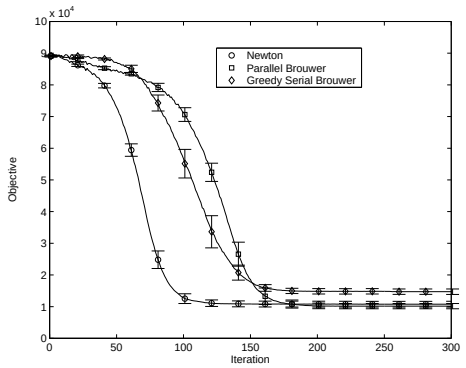

Figure 1.1: Comparison of Lagrangian minimization techniques for the bin packing problem.

A more efficient alternative is to delegate the agent to be updated at the start of each iteration - either randomly or sequentially. Now only a single agent needs to evaluate its private utility, reducing the number of function calls by a factor of (N+1)/2. This variant is compared in Figure 1.2 on a function call rather than iteration basis. The agents were updated in sequence although random selection obtained similar results. For comparison Nearest Newton is shown with two different Monte-Carlo block sizes: 100 and 20. Using Nearest Newton with a reduced block size is seen to be more effective at lowering the number of function calls, although the sequential serial Brouwer does manage to find a comparable final objective. The initial behavior of the serial Brouwer is again the result of limited sampling and overly aggressive updating. The slower convergence results from expending Monte-Carlo samples, and the associated function calls, on updating agents which are not important to the reducing the Lagrangian. This arises from the current implementation which updates each agent every N iterations. Variants which estimate which agent to update are likely to be more efficient.

1.3.2 Continuous optimization example problem

To illustrate the Lagrangian minimization techniques in the continuous domain, a classical calculus of variations problem is solved, the Brachistochrone problem [5]. In this problem the objective is to find the minimum time trajectory between two points for an object moving only under the influence of gravity. Following [5] the objective function is:

$$t_{12} = \int_{x_1,y_1}^{x_2,y_2} f \, dx$$

where

$$f = (1 + (dy/dx)^2)^{1/2} \, (2gy)^{1/2} \tag{1.4}$$

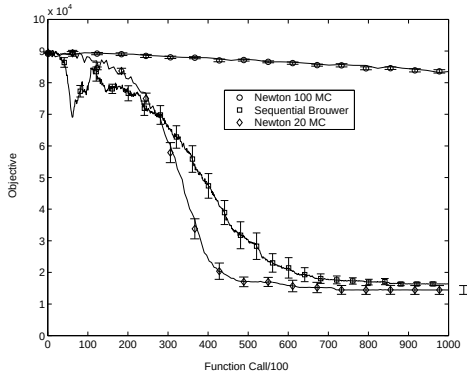

Figure 1.2: Comparison of modified Lagrangian minimization techniques on a function call basis for the bin packing problem.

To cast the problem as an optimization with respect to the vertical location of points along the trajectory both the integral and the derivative are approximated. A trapezoidal approximation is made to the integral at N points and a central finite difference is used for the derivative. The result is the following optimization problem,

$$\min_{\vec{y}} G = \frac{\Delta x}{2}\left[f_0 + 2f_1 + ... + 2f_N - 1 + 2f_N\right]$$

where, for the interior points

$$f_i = (1 + [\frac{1}{2\Delta x}(y_{i+1} - y_{i-1}]^2)^{1/2}\ (2gy_i)^{1/2} \tag{1.5}$$

For the boundary points, f_0 and f_N, forward or backward approximations are used for the derivatives.

This optimization problem was solved by a commercially available gradient based optimizer [6] and by the Lagrangian minimization approaches described in Section 1.2. The approaches described in Section 1.2 are particularly applicable to objectives such as Eq. 1.5 due to sparse nature of the interactions between the variables (agents). The contributions to the objective are functions only of a single variable and that variables neighbors. This leads directly to a private utility, for the interior points, of the form,

$$\begin{aligned} g_i(y_{i-1}, y_i, y_{i+1}) = \frac{\Delta x}{2}\big[&2f_{i-1}(y_{i-2}, y_{i-1}, y_i) \\ &+2f_i(y_{i-1}, y_i, y_{i+1}) \\ &+2f_{i+1}(y_i, y_{i+1}, y_{i+2})\big] \end{aligned} \tag{1.6}$$

Similar private utilities can be obtained for the first and last nodes. Note this private utility has no bias since it includes all the dependencies of the world

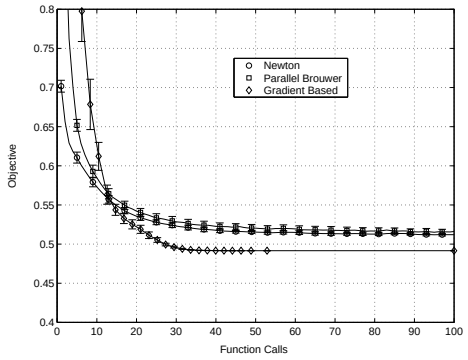

Figure 1.3: Comparison of Lagrangian minimization approaches using sampling for the Brachistochrone problem.

utility (objective function) upon agent i. Since the dependencies are known, only the other agents' probabilities indicated in 1.6 are required to evaluate the expectations. This suggests two possible variants of the techniques described in Section 1.2. First, sampling could be used as before but in place of the Wonderful Life Utility, use Eq. 1.6. Second, the probability distributions can be integrated along with Eq. 1.6 to obtain the expected private utilities. For either approach the probability updating is the same, only the approach for obtaining the expected utilities differs. Also note that second variant is now deterministic since no sampling is involved.

Figure 1.3 shows the convergence history of the objective function for the gradient based, parallel Brouwer and Newton approaches. The serial Brouwer result is not shown since there is no longer any efficiency advantage given the easily calculated private utility. Relevant parameters are $\alpha = 0.2$, $\gamma = 0.8$, 10 Monte-Carlo samples, and $T = 0.01$. Also, the starting point is set to (0,0) and the ending point to (1,1). In all cases, the optimizations were performed 10 times and the 90% confidence bars are shown. For the gradient based optimization a random starting point was used each time. The sampled approaches (Newton and Brouwer) perform comparably, finding a minimum about 5% higher than the gradient based optimum. The key result is comparable convergence rate to the gradient based approach in terms of function calls.

Figure 1.4 shows converged probability distributions for the agents versus their position from the analytical case. This illustrates the additional information provided by the non-gradient based approaches approaches: non-linear sensitivities. At convergence the probability distributions are related to the objective through the Boltzmann distribution. As a result, sensitivities of the optimum with respect to each variable are provided without additional computational cost.

Figure 1.5 shows the iteration history for the analytical case. The upper plot shows the objective function, while the lower plot shows the convergence

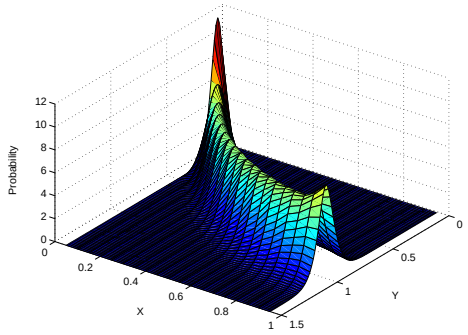

Figure 1.4: Converged probability distribution for the Brachistochrone problem.

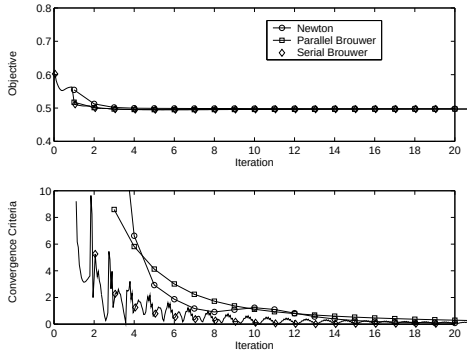

Figure 1.5: Comparison of Lagrangian minimization approaches for analytical version of the Brachistochrone problem.

criteria. While all three quickly find a good minimum, serial Brouwer is actually converging faster. This makes sense, especially for the sequential ordering, the agents can use probabilities from their neighbors just after they have been updated.

1.4 Conclusions

Product Distribution (PD) theory provides an effective framework for performing distributed optimization. One theoretical perspective for PD theory has been provided and a distributed optimization algorithm based upon it has been developed. The approach has been demonstrated on two example problems, one discrete and one continuous. The comparisons included results obtained with variants of distributed Reinforcement Learning (RL) inspired optimization approaches. The inter-relationship between the approaches has been highlighted.

Bibliography

[1] Wolpert, D. H., Tumer, K., "Collective Intelligence, Data Routing, and Braess' Paradox," Journal of Artificial Intelligence Research, 2002.

[2] Wolpert, D.H., Wheeler, K., Tumer, K., "Collective Intelligence for Control of Distributed Dynamical Systems," Europhysics Letters, vol. 49 issue 6, 708-714, 2000.

[3] Wolpert, D.H., "Information theory - the bridge connecting bounded rational game theory and statistical physics", in Complex Engineering Systems, D. Braha, Ali Minai, and Y. Bar-Yam (Editors), Perseus books, in press.

[4] J.E.Beasley, "OR-Library: distributing test problems by electronic mail", Journal of the Operational Research Society, 41 (11), pp 1069-1072, 1990. http://mscmga.ms.ic.ac.uk/info.html

[5] Eric W. Weisstein, "Brachistochrone Problem." From *MathWorld* – A Wolfram Web Resource. http://mathworld.wolfram.com/BrachistochroneProblem.html

[6] *MATLAB Optimization Toolbox User's Guide, Version 2.* The MathWorks, 2000.

[7] Duda, R., Hart, P., and Stork, D., *Pattern Recognition*, Wiley, 2001.

[8] D. Fudenburg and D. K. Levine, *The Theory of Learning in Games*, MIT Press, Cambridge, MA, 1998.